A statistical machine translation primer

Cancedda, Nicola; Dymetman, Marc; Foster, George; Goutte, Cyril

Learning Machine Translation, Neural Information Processing, pp. 1-37, 2009

1 A Statistical Machine Translation Primer

Nicola Cancedda
Marc Dymetman
George Foster
Cyril Goutte

This first chapter is a short introduction to the main aspects of statistical machine translation (SMT). In particular, we cover the issues of automatic evaluation of machine translation output, language modeling, word-based and phrase-based translation models, and the use of syntax in machine translation. We will also do a quick roundup of some more recent directions that we believe may gain importance in the future. We situate statistical machine translation in the general context of machine learning research, and put the emphasis on similarities and differences with standard machine learning problems and practice.

1.1 Background

Machine translation (MT) has a long history of ambitious goals and unfulfilled promises. Early work in automatic, or "mechanical" translation, as it was known at the time, goes back at least to the 1940s. Its progress has, in many ways, followed and been fueled by advances in computer science and artificial intelligence, despite a few stumbling blocks like the ALPAC report in the United States (Hutchins, 2003).

Availability of greater computing power has made access to and usage of MT more straightforward. Machine translation has also gained wider exposure to the public through several dedicated services, typically available through search engine services. Most internet users will be familiar with at least one of Babel Fish,[1] Google Language Tools,[2] or Windows Live Translator.[3] Most of these services used to be

1. `http://babelfish.yahoo.com/` or `http://babelfish.altavista.com/`
2. `http://www.google.com/language_tools`
3. `http://translator.live.com/`

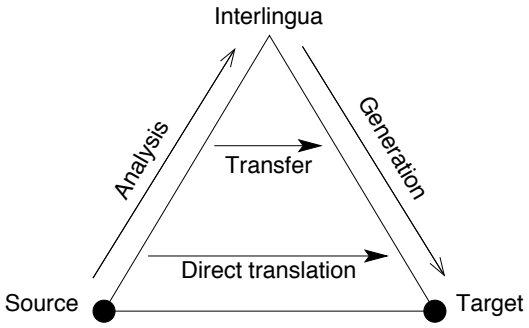

Figure 1.1 The machine translation pyramid. Approaches vary depending on how much analysis and generation is needed. The *interlingua* approach does full analysis and generation, whereas the *direct translation* approach does a minimum of analysis and generation. The *transfer* approach is somewhere in between.

powered by the rule-based system developped by Systran.[4] However, some of them (e.g., Google and Microsoft) now use statistical approaches, at least in part.

In this introduction and the rest of this book, translation is defined as the task of transforming an existing text written in a *source language*, into an equivalent text in a different language, the *target language*. Traditional MT (which in the context of this primer, we take as meaning "prestatistical") relied on various levels of linguistic analysis on the source side and language generation on the target side (see figure 1.1).

The first statistical approach to MT was pioneered by a group of researchers from IBM in the late 1980s (Brown et al., 1990). This may in fact be seen as part of a general move in computational linguistics: Within about a decade, statistical approaches became overwhelmingly dominant in the field, as shown, for example, in the proceedings of the annual conference of the Association for Computational Linguistics (ACL).

The general setting of statistical machine translation is to learn how to translate from a large corpus of pairs of equivalent source and target sentences. This is typically a machine learning framework: we have an *input* (the source sentence), an *output* (the target sentence), and a model trying to produce the correct output for each given input.

There are a number of key issues, however, some of them specific to the MT application. One crucial issue is the evaluation of translation quality. Machine learning techniques typically rely on some kind of cost optimization in order to learn relationships between the input and output data. However, evaluating automatically the quality of a translation, or the cost associated with a given MT output, is a very hard problem. It may be subsumed in the broader issue of language understanding, and will therefore, in all likelihood, stay unresolved for some time. The difficulties

4. `http://www.systransoft.com/`

associated with defining and automatically calculating an evaluation cost for MT will be addressed in section 1.2.

The early approach to SMT advocated by the IBM group relies on the *source-channel* approach. This is essentially a framework for combining two models: a *word-based* translation model (section 1.3) and a *language model* (section 1.4). The translation model ensures that the system produces target hypotheses that correspond to the source sentence, while the language model ensures that the output is as grammatical and fluent as possible.

Some progress was made with word-based translation models. However, a significant breakthrough was obtained by switching to *log-linear models* and *phrase-based* translation. This is described in more detail in section 1.5.

Although the early SMT models essentially ignored linguistic aspects, a number of efforts have attempted to reintroduce linguistic considerations into either the translation or the language models. This will be covered in section 1.6 and in some of the contributed chapters later on. In addition, we do a quick overview of some of the current trends in statistical machine translation in section 1.7, some of which are also addressed in later chapters.

Finally, we close this introductory chapter with a discussion of the relationships between machine translation and machine learning (section 1.8). We will address the issue of positioning translation as a learning problem, but also issues related to optimization and the problem of learning from an imprecise or unavailable loss.

1.2 Evaluation of Machine Translation

Entire books have been devoted to discussing what makes a translation a good translation. Relevant factors range from whether translation should convey emotion as well and above meaning, to more down-to-earth questions like the intended use of the translation itself.

Restricting our attention to machine translation, there are at least three different tasks which require a quantitative measure of quality:

1. assessing whether the output of an MT system can be useful for a specific application (*absolute evaluation*);

2. (a) comparing systems with one another, or similarly (b) assessing the impact of changes inside a system (*relative evaluation*);

3. in the case of systems based on learning, providing a loss function to guide parameter tuning.

Depending on the task, it can be more or less useful or practical to require a human intervention in the evaluation process. On the one hand, humans can rely on extensive language and world knowledge, and their judgment of quality tends to be more accurate than any automatic measure. On the other hand, human judgments tend to be highly subjective, and have been shown to vary considerably between

different judges, and even between different evaluations produced by the same judge at different times.

Whatever one's position is concerning the relative merits of human and automatic measures, there are contexts—such as (2(b)) and (3)—where requiring human evaluation is simply impractical because too expensive or time-consuming. In such contexts fully automatic measures are necessary.

A good automatic measure should above all correlate well with the quality of a translation as it is perceived by human readers. The ranking of different systems given by such a measure (on a given sample from a given distribution) can then be reliably used as a proxy for the ranking humans would produce. Additionally, a good measure should also display low intrasystem variance (similar scores for the same system when, e.g., changing samples from the same dataset, or changing human reference translations for the same sample) and high intersystem variance (to reliably discriminate between systems with similar performance). If those criteria are met, then it becomes meaningful to compare scores of different systems on different samples from the same distribution.

Correlation with human judgment is often assessed based on collections of (human and automatic) translations manually scored with *adequacy* and *fluency* marks on a scale from 1 to 5. Adequacy indicates the extent to which the information contained in one or more reference translations is also present in the translation under examination, whereas fluency measures how grammatical and natural the translation is. An alternate metric is the direct test of a user's comprehension of the source text, based on its translation (Jones et al., 2006).

A fairly large number of automatic measures have been proposed, as we will see, and automatic evaluation has become an active research topic in itself. In many cases new measures are justified in terms of correlation with human judgment. Many of the measures that we will briefly describe below can reach Pearson correlation coefficients in the 90% region on the task of ranking systems using a few hundred translated sentences. Such a high correlation led to the adoption of some such measures (e.g., BLEU and NIST scores) by government bodies running comparative technology evaluations, which in turn explains their broad diffusion in the research community. The dominant approach to perform model parameter tuning in current SMT systems is "minimum error–rate training" (MERT; see section 1.5.3), where an automatic measure is explicitly optimized.

It is important to notice at this point that high correlation was demonstrated for existing measures only *at the system level*: when it comes to the score given to individual translated sentences, the Pearson correlation coefficient between automatic measures and human assessments of adequacy and fluency drops to 0.3 to 0.5 (see, e.g., Banerjee and Lavie, 2005; Leusch et al., 2005). As a matter of fact, even the correlation between human judgments decreases drastically. This is an important observation in the context of this book, because many machine learning algorithms require the loss function to decompose over individual inferences/translations. Unlike in many other applications, when dealing with machine translation loss functions that decompose over inferences are only pale indicators of quality as perceived

by users. While in document categorization it is totally reasonable to penalize the number of misclassified documents, and the agreement between the system decision on a single document and its manually assigned label is a very good indicator of the perceived performance of the system on that document, an automatic score computed on an individual sentence translation is a much less reliable indicator of what a human would think of it.

Assessing the quality of a translation is a very difficult task even for humans, as witnessed by the relatively low interannotator agreement even when *quality* is decomposed into *adequacy* and *fluency*. For this reason most automatic measures actually evaluate something different, sometimes called *human likeness*. For each source sentence in a test set a reference translation produced by a human is made available, and the measure assesses how similar the translation proposed by a system is to the reference translation. Ideally, one would like to measure how similar the meaning of the proposed translation is to the meaning of the reference translation: an ideal measure should be invariant with respect to sentence transformations that leave meaning unchanged (paraphrases). One source sentence can have many perfectly valid translations. However, most measures compare sentences based on superficial features which can be extracted very reliably, such as the presence or absence in the references of n-grams from the proposed translation. As a consequence, these measures are far from being invariant with respect to paraphrasing. In order to compensate for this problem, at least in part, most measures allow considering more than one reference translation. This has the effect of improving the correlation with human judgment, although it imposes on the evaluator the additional burden of providing multiple reference translations.

In the following we will briefly present the most widespread automatic evaluation metrics, referring to the literature for further details.

1.2.1 Levenshtein-Based Measures

A first group of measures is inherited from speech recognition and is based on computing the edit distance between the candidate translation and the reference. This distance can be computed using simple dynamic programming algorithms.

Word error rate (WER) (Nießen et al., 2000) is the sum of insertions, deletions, and substitutions normalized by the length of the reference sentence. A slight variant (WERg) normalizes this value by the length of the Levenshtein path, i.e., the sum of insertions, deletions, substitutions, and matches: this ensures that the measure is between zero (when the produced sentence is identical to the reference) and one (when the candidate must be entirely deleted, and all words in the reference must be inserted).

Position-independent word error rate (PER) (Tillmann et al., 1997b) is a variant that does not take into account the relative position of words: it simply computes the size of the intersection of the bags of words of the candidate and the reference, seen as multi-sets, and normalizes it by the size of the bag of words of the reference.

A large U.S. government project called "Global Autonomous Language Exploitation" (GALE) introduced another variant called the *translation edit rate (TER)*(Snover et al., 2006). Similarly to WER, TER counts the minimal number of insertion, deletions, and substitutions, but unlike WER it introduces a further unit-cost operation, called a "shift," which moves a whole substring from one place to another in the sentence.

In the same project a further semiautomatic *human-targeted translation edit rate (HTER)* is also used. While WER and TER only consider a pre-defined set of references, and compare candidates to them, in computing HTER a human is instructed to perform the minimal number of operations to turn the candidate translation into a grammatical and fluent sentence that conveys the same meaning as the references. Not surprisingly, Snover et al. (2006) show that HTER correlates with human judgments considerably better than TER, BLEU, and METEOR (see below), which are fully automatic.

1.2.2 N-Gram–Based Measures

A second group of measures, by far the most widespread, is based on notions derived from information retrieval, applied to the n-grams of different length that appear in the candidate translation. In particular, the basic element is the *clipped n-gram precision*, i.e., the fraction of n-grams in a set of translated sentences that can be found in the respective references.[5]

BLEU (Papineni et al., 2002) is the geometric mean of clipped n-gram precisions for different n-gram lengths (usually from one to four), multiplied by a factor (*brevity penalty*) that penalizes producing short sentences containing only highly reliable portions of the translation.

BLEU was the starting point for a measure that was used in evaluations organized by the U.S. National Institute for Standards and Technology (NIST), and is thereafter referred to as the *NIST* score (Doddington, 2002). NIST is the arithmetic mean of clipped n-gram precisions for different n-gram lengths, also multiplied by a (different) brevity penalty. Also, when computing the NIST score, n-grams are weighted according to their frequency, so that less frequent (and thus more informative) n-grams are given more weight.

1.2.3 The Importance of Recall

BLEU and NIST are forced to include a brevity penalty because they are based only on n-gram precision. N-gram recall was not introduced because it was not immediately obvious how to meaningfully define it in cases where multiple reference

5. Precision is *clipped* because counts are thresholded to the number of occurrences of n-grams in the reference, so that each n-gram occurrence in the reference can be used to "match" at most one n-gram occurrence in the proposed sentence. Note also that the precision is computed for all n-grams in a document at once, not sentence by sentence.

translations are available. A way to do so was presented in Melamed et al. (2003): the *general text matcher* (GTM) measure relies on first finding a *maximum matching* between a candidate translation and a set of references, and then computing the ratio between the size of this matching (modified to favor long matching contiguous n-grams) and the length of the translation (for precision) or the mean length of the reference (for recall). The harmonic mean of precision and recall can furthermore be taken to provide the *F-measure*, familiar in natural language processing. Two very similar measures are ROUGE-L and ROUGE-W, derived from automatic quality measures used for assessing document summaries, and extended to MT (Lin and Och, 2004a). ROUGE-S, introduced in the same paper, computes precision, recall, and F-measure based on *skip-bigram* statistics, i.e., on the number of bigrams possibly interrupted by gaps.

A further measure, which can be seen as a generalization of both BLEU and ROUGE (both -L and -S), is BLANC (Lita et al., 2005). In BLANC the score is computed as a weighted sum of all matches of all subsequences (i.e., n-grams possibly interrupted by gaps) between the candidate translation and the reference. Parameters of the scoring function can be tuned on corpora for which human judgments are available in order to improve correlation with adequacy, fluency, or any other measure that is deemed relevant.

Finally, the proposers of *METEOR* (Banerjee and Lavie, 2005) put more weight on recall than on precision in the harmonic mean, as they observed that this improved correlation with human judgment. METEOR also allows matching words which are not identical, based on stemming and possibly on additional linguistic processing.

1.2.4 Measures Using Syntax

Liu and Gildea (2005) propose a set of measures capable of taking long-distance syntactic phenomena into account. These measures require the candidates and the references to be syntactically analyzed. Inspired by BLEU and NIST, averaged precision of paths or subtrees in the syntax trees are then computed. In the same line, Giménez and Màrquez (2007b) also use linguistic processing, up to shallow semantic analysis, to extract additional statistics that are integrated in new measures.

While these measures have the drawback of requiring the availability of an accurate and robust parser for the target language, and of making the measure dependent on the selected parser, the authors show significantly improved correlation with human judgments of quality.

1.2.5 Evaluating and Combining Measures

Measures of translation quality are usually themselves evaluated according to Pearson (and less often Spearman) correlation coefficients with human judgments on some test set. Lin and Och (2004b) observe that this criterion is not stable

across data sets. They thus propose an alternative metameasure, *ORANGE*, based on the additional assumption that a good measure should tend to rank reference translations higher than machine translations. Using a machine translation system, an *n–best* list of candidate translations is generated. Each element in the list is then scored using the measure of interest against a set of m reference translations. Reference translations themselves are scored using the same measure, and a global ranking is established. From this ranking, it is possible to compute the average rank of reference translations. Averaging this average rank across all sentences in the test set provides the ORANGE score. This score is then shown to be more consistent than correlation coefficients in ranking evaluation measures on data produced by a single MT system on a given test corpus.

An interesting method to combine the complementary strengths of different measures, and at the same time evaluate evaluation measures and estimate the reliability of a test set, is QARLA (Giménez and Amigó, 2006).

1.2.6 Statistical Significance Tests

Whatever automatic measure one uses, tests of statistical significance provparametric methods are usually considered better suited to the task, especially bootstrap resampling and approximate randomization. Riezler and Maxwell (2005) provide a good discussion of these tests in the context of machine translation evaluation.

1.3 Word-Based MT

Word-based statistical MT originated with the classic work of Brown et al. (1993). Given a source sentence $\mathbf{f}$, Brown et al. seek a translation $\hat{\mathbf{e}}$ defined by the "fundamental equation of statistical MT":

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, p(\mathbf{f}|\mathbf{e})\, p(\mathbf{e}). \tag{1.1}$$

Here the conditional distribution $p(\mathbf{e}|\mathbf{f})$ is decomposed into a *translation model* $p(\mathbf{f}|\mathbf{e})$ and a *language model* $p(\mathbf{e})$. By analogy with cryptography or communication theory, this is sometimes referred to as a *source-channel* (or *noisy-channel*) model, where $p(\mathbf{e})$ is a known "source" distribution, $p(\mathbf{f}|\mathbf{e})$ is a model of the process that encodes (or corrupts) it into the observed sentence $\mathbf{f}$, and the argmax is a *decoding* operation. This decomposition has the advantage of simplifying the design of $p(\mathbf{f}|\mathbf{e})$ by factoring out responsibility for ensuring that $\mathbf{e}$ is well formed into the language model $p(\mathbf{e})$ (language models will be covered in more detail in section 1.4). It also allows the language model to be trained separately on monolingual corpora, which are typically more abundant than parallel corpora.

Brown et al. elaborate a series of five generative models (numbered 1 through 5) for $p(\mathbf{f}|\mathbf{e})$ which are known collectively as the *IBM* models. Each model in the series improves on its predecessor by adding or reinterpreting parameters. The

models are trained to maximize the likelihood of a parallel corpus seen as a set of statistically independent sentence pairs,[6] with the earlier models used to provide initial parameter estimates for later models. Early stopping during expectation–maximization (EM) training is typically used to avoid overfitting.

The IBM models are defined over a hidden *alignment* variable $\mathbf{a}$ which captures word-level correspondences between $\mathbf{f}$ and $\mathbf{e}$:

$$p(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} p(\mathbf{f}, \mathbf{a}|\mathbf{e})$$

where $\mathbf{a}$ is a vector of alignment positions a_j for each word f_j in $\mathbf{f} = f_1 \dots f_J$. Each a_j takes on a value i in $[1, I]$ to indicate a connection to word e_i in $\mathbf{e} - e_1 \dots e_I$, or is 0 to indicate a null connection. Note that this scheme is asymmetrical: words in $\mathbf{f}$ may have at most a single connection, while words in $\mathbf{e}$ may have from 0 to J connections. This asymmetry greatly reduces the number of alignments which must be considered, from 2^{IJ} if arbitrary connections are allowed, to $(I+1)^J$.

1.3.1 Models 1, 2, and HMM

IBM models 1 and 2, as well as the commonly used *HMM* variant due to Vogel et al. (1996), are based on the following decomposition[7] of $p(\mathbf{f}, \mathbf{a}|\mathbf{e})$:

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) \approx \prod_{j=1}^{J} p(f_j|e_{a_j}) p(a_j|a_{j-1}, j, I, J).$$

These three models share the family of *lexical translation parameters* $p(f|e)$, but differ in how they parameterize alignments:

$$p(a_j|a_{j-1}, j, I, J) = \begin{cases} 1/(I+1) & \text{IBM 1} \\ p(a_j|j, I, J) & \text{IBM 2} \\ p(a_j - a_{j-1}) & \text{HMM} \end{cases},$$

i.e., in IBM 1 all connections are equally likely, in IBM 2 they depend on the absolute positions of the words being connected, and in the HMM model they depend on the displacement from the previous connection. Och and Ney (2003) discuss how to extend the HMM model to handle null connections. Maximum likelihood training using the EM algorithm is straightforward for all three models; for IBM 1 it is guaranteed to find a global maximum (Brown et al., 1993), making this model a good choice for initializing the lexical parameters. Due to the large number of lexical

6. A necessary prerequisite is identifying translated sentence pairs in parallel documents. This is nontrivial in principle, but in practice fairly simple methods based on sentence length and surface lexical cues, e.g., Simard et al. (1992), are often adequate. For more difficult corpora, a bootstrapping approach (Moore, 2002) can be used.
7. Omitting a factor for normalizing across all sentence lengths J.

parameters, it is common practice to prune out word pairs (f,e) whose probability $p(f|e)$ falls below a certain threshold after IBM1 training.

1.3.2 Models 3, 4, and 5

IBM 1/2 and HMM are based on a generative process in which each word in $\mathbf{f}$ is filled in (from left to right) by first choosing a connecting position in $\mathbf{e}$ according to the position's alignment probability, then choosing the word's identity according to its translation probability given the connected target word. In the more complex models 3, 4, and 5, the emphasis of the generative process shifts to the target sentence: for each word in $\mathbf{e}$, first the number of connected words is chosen (its *fertility*), then the identities of these words, and finally their positions in $\mathbf{f}$. These models retain word-for-word translation parameters $p(f|e)$, as well as asymmetrical alignments in which each source word may connect to at most one target word.

Model 3 incorporates a set of fertility parameters $p(\phi|\mathbf{e})$, where ϕ is the number of words connected to $\mathbf{e}$, and reinterprets model 2's alignment parameters as *distortion* parameters $p(j|i, I, J)$.

Model 4 replaces model 3's distortion parameters with ones designed to model the way the set of source words generated by a single target word tends to behave as a unit for the purpose of assigning positions. The first word in the ith unit is assigned to position j in the source sentence with probability $p(j - \overline{U_{i-1}}|e_i, f_j)$, where $\overline{U_{i-1}}$ is the average position of the words in the $(i-1)$th unit.[8] Subsequent words in the ith unit are placed with probability $p(j - U_{i,j-1}|f_j)$, where $U_{i,j-1}$ gives the position of the $(j-1)$th word in this unit.

Models 3 and 4 are *deficient* (nonnormalized) because their generative processes may assign more than one source word to the same position. Model 5 is a technical adjustment to correct for this problem.

The EM algorithm is intractable for models 3, 4, and 5 because of the expense of summing over all alignments to calculate expectations. The exact sum is therefore approximated by summing over a small set of highly probable alignments, each of whose probability can be calculated efficiently. Model 2 Viterbi alignments are used to initialize the set, which is then expanded using a greedy perturbative search involving moving or swapping individual links.

1.3.3 Search

The problem of searching for an optimal translation (the argmax operation in Eq. (1.1)) is a difficult one for the IBM models. Roughly speaking, there are two sources of complexity: finding the best bag of target words according to the many-to-one source-target mapping implicit in $p(\mathbf{f}|\mathbf{e})$; and finding the best target word

8. To combat data sparseness, Brown et al. map e_i and f_j in this expression to one of 50 equivalence classes defined over source and target vocabularies.

order according to $p(\mathbf{e})$. Knight (1999) shows that decoding is in fact NP-complete for the IBM models through separate reductions exploiting each of these sources of complexity. However, in practice, heuristic techniques work quite well. Germann et al. (2001) describe a Viterbi stack-based algorithm that operates quickly and makes relatively few search errors, at least on short sentences. As this algorithm is similar to search algorithms used with phrase-based translation models, we defer a description to section 1.5.

1.3.4 Current Status

The IBM models have been supplanted by the more recent phrase-based approach to SMT, described in section 1.5, which is conceptually simpler and produces better results. However, they retain a central role due to their ability to produce good word alignments, which are a key ingredient in training phrase-based models. Despite significant recent attention to the problem of word alignment for this and other purposes, IBM 4 alignments—typically produced using the GIZA++ toolkit (see appendix), and symmetrized using the method of Och and Ney (2000a)—remain the most often-used baseline for work in this area.

Unlike the phrase-based model and the later IBM models, models 1/2 and HMM also allow efficient computation of a smooth conditional distribution $p(\mathbf{f}|\mathbf{e})$ over bilingual sentence pairs. This makes them well suited for applications requiring analysis of existing sentence pairs, such as cross-language information retrieval.

1.4 Language Models

A language model (LM), in the basic sense of the term, is a computable probability distribution over word sequences, typically sentences, which attempts to approximate an underlying stochastic process on the basis of an observed corpus of sequences produced by that process.

Language models have many applications apart from statistical machine translation, among them: speech recognition (SR), spelling correction, handwriting recognition, optical character recognition, information retrieval. Historically, much of their development has been linked to speech recognition and often the methods developed in this context have been transposed to other areas; to a large extent this remains true today.

According to the dominant "generative"paradigm in language modeling (and to our definition above), developing a language model should actually only depend on a corpus of texts, not on the application context. The standard measure of adequacy of the language model is then its *perplexity*,[9] an information-theoretic quantity that

9. If $LL_p(T) = log_2\, p(T)$ represents the log-likelihood of the test corpus T relative to the model p, then the perplexity of p on T is defined as $2^{-LL_p(T)}$.

measures the cost of coding a test corpus with the model, and which is provably minimized when the model represents the "true" distribution of the underlying stochastic process.

Recently, some work has started to challenge the dominant paradigm, in an approach known as "discriminative" or "corrective" language modeling, where the focus is more on minimizing errors in the context of a specific application, a criterion that, due to inevitable inadequacies in the application-dependent aspects of the overall probabilistic model (such as the "acoustic model" in SR, or the "translation model" in SMT), does not coincide with perplexity minimization.

This section will mostly focus on the generative paradigm, and will give some pointers to discriminative approaches. Good general references on language models are Goodman (2001) and Rosenfeld (2000), as well as the tutorial of Charniak and Goodman (2002), which have influenced parts of this section.

1.4.1 N-Gram Models and Smoothing Techniques

Still by far the dominant technique for language modeling is the n-gram approach, where the probability of a sequence of words $w_1, w_2, \ldots, w_m$ is approximated, using the case n=3 (trigram) as an illustration, as

$$p(w_1, w_2, \ldots, w_m) \approx \prod_i p(w_i|w_{i-2}, w_{i-1}).$$

The central issue in such models is how to estimate the conditional probabilities $p(w_i|w_{i-2}, w_{i-1})$ from the corpus. The simplest way, maximum likelihood, corresponds to estimating these probabilities as a ratio of counts in the corpus (where # indicates a count of occurrences):

$$p(w_i|w_{i-2}, w_{i-1}) = \frac{\#(w_{i-2}, w_{i-1}, w_i)}{\#(w_{i-2}, w_{i-1})},$$

but this approach suffers from obvious "overfitting" defects; in particular the model assigns a zero probability to a trigram which has not been observed in the corpus. In order to address this problem, several "smoothing" techniques have been devised, which can be roughly characterized by three central representatives.

In *Jelinek-Mercer* interpolated smoothing, one writes

$$p_{JM}(w_i|w_{i-2}, w_{i-1}) = \lambda_3 \frac{\#(w_{i-2}, w_{i-1}, w_i)}{\#(w_{i-2}, w_{i-1})} + \lambda_2 \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} + \lambda_1 \frac{\#(w_i)}{\#(\bullet)},$$

which corresponds to interpolating trigram, bigram, and unigram estimates, and where the λ weights are tuned through cross-validation on the corpus. In refined versions of this approach the weights may be tuned differently depending on different ranges of frequencies of the corresponding denominators.

In *Katz backoff* smoothing, the general idea is that when the trigram counts are low, one will *back off* to the bigram estimates:

$$p_K(w_i|w_{i-2},w_{i-1}) = \begin{cases} \frac{\#^*(w_{i-2},w_{i-1},w_i)}{\#(w_{i-2},w_{i-1})} & \text{if } \#(w_{i-2},w_{i-1},w_i) > 0 \\ \lambda\ p_K(w_i|w_{i-1}) > 0 & \text{otherwise} \end{cases},$$

where $\#^*(w_{i-2},w_{i-1},w_i)$ is a "discounted" count according to the Good-Turing formula, which has the effect of displacing some mass from observed events to unobserved events (and which is based on the insight that the number of types[10] which are observed once in the corpus is indicative of the number of unobserved types which are "just waiting there" to be observed, for which some mass should be reserved), and where λ is a normalization factor that weighs the influence of the *backoff* to bigram model.

In *Kneser-Ney backoff* smoothing, and for expository reasons considering only bigram models here, one writes

$$p_{KN}(w_i|w_{i-1}) = \begin{cases} \frac{\#(w_{i-1},w_i)-D}{\#(w_{i-1})} & \text{if } \#(w_{i-1},w_i) > 0 \\ \lambda \frac{\tilde{\#}(\bullet,w_i)}{\tilde{\#}(\bullet,\bullet)} & \text{otherwise} \end{cases},$$

where $D \in [0,1]$ is a fixed discounting factor, λ is a normalization factor, and where (our notation/reconstruction) $\tilde{\#}(\bullet,\bullet)$ (resp. $\tilde{\#}(\bullet,w_i)$) is the number of different bigram *types* (resp. bigram types ending in w_i) found in the corpus. Thus, a crucial difference between Kneser-Ney and other techniques is that it does not back off to a quantity that measures the relative frequency $\frac{\#(w_i)}{\#(\bullet)}$ of *occurrences* of the word w_i, which can also be written in the form $\frac{\#(\bullet,w_i)}{\#(\bullet,\bullet)} = \frac{\#(w_i)}{\#(\bullet)}$, but to a quantity $\frac{\tilde{\#}(\bullet,w_i)}{\tilde{\#}(\bullet,\bullet)}$ that measures the "context-type unspecificity" of w_i in terms of the number of different word *types* which may precede w_i. The intuition is that the less context type–specific (i.e., more context type–unspecific) w_i is, the more we would expect to recognize it in a context w_{i-1}, w_i we have never witnessed before.

It is probably fair to say that n-gram with Kneser-Ney smoothing is currently the most widely accepted language modeling technique in practice, sometimes even applied to 4-gram or 5-gram modeling when large enough corpora are available.

Caching

N-gram models are severely limited in their ability to account for nonlocal statistical dependencies. One simple and efficient technique allowing use of the nonlocal context is caching: remembering words that have been produced in the recent history, for example during a dictation session, and predicting that such words have a tendency to repeat later in the session (Kuhn and de Mori, 1990). In its simplest

10. *Types* correspond to classes of objects, as opposed to *tokens*, which correspond to occurrences of these classes. For instance, there are two tokens of the type "man" in the expression "man is a wolf to man."

form, this consists in interpolating a standard trigram model with a unigram cache model $p_{cache}(w_i|w_1,\ldots w_{i-1}) = (i-1)^{-1}(\#\text{instances of } w_i \text{ in } w_1,\ldots w_{i-1})$, but variants exist which consider instances of bigrams or trigrams in the history. One potential problem with caching is that recognition errors early in the history may have a snowball effect later on, unless the history is guaranteed to be accurate, such as in an interactive dictation environment in which the user validates the system outputs.

Class-Based Smoothing

Rather than using words as the basis for n-gram smoothing as discussed so far, another option is to first group words into classes that exhibit similar linguistic behavior, then to use these classes to model statistical dependencies. A simple example of this approach is the following:

$$p(w_i|w_{i-2},w_{i-1}) \approx p(w_i|C_i)\, p(C_i|w_{i-2},w_{i-1}),$$

where C_i is the class associated with w_i . The point of this model is that the classes have higher corpus frequencies than the individual words, and therefore conditional probabilities involving classes can be more reliably estimated on the basis of training data. There are many variants of this basic idea, along three main dimensions: (i) the classes may appear in diverse combinations on the left or right side of the conditioning sign; (ii) the association of a class to a word may be hard, with one class per word (equation shown), or soft, with several classes per word (in this case the equation shown needs to include a sum over classes); (iii) the classes may be associated with the words according to predefined categorization schemes, for instance part-of-speech tags or predefined semantic categories; the last case is especially useful for restricted target domains, for instance speech recognition for air travel reservations. At the opposite end of the spectrum, the classes may be data-driven and obtained through various clustering techniques, a criterion of a good clustering being a low perplexity of the corresponding language model.

One especially interesting application of classes is their possible use for modeling languages with a richer morphology than English, for instance by taking a class to be a lemma or a part of speech or by combining both aspects (Maltese and Mancini, 1992; El-Bèze and Derouault, 1990). Recent approaches to factored translation and language models (see sections 1.4.3 and 1.7.1) work in a similar spirit.

1.4.2 Maximum Entropy Models

Maximum entropy models (aka log-linear models), have been an important tool of statistical natural language processing (NLP) since the early 1990s, in particular in the context of statistical machine translation as we will see in the next section, but also for language modeling proper (Rosenfeld, 2000; Jelinek, 1998), where their role is more controversial.

For language modeling, these models come in two flavors. The main one, which we will call *history-based maxent models*, will be discussed first, then we will briefly discuss so-called *whole-sentence maxent models*.

History-Based Maximum Entropy Models

Generally speaking, history-based language models are models of the form

$$p(w_1, w_2, \ldots, w_m) = \prod_i p(w_i|h_i),$$

where $h_i = w_1, \ldots, w_{i-2}, w_{i-1}$ is the history, and where $p(w_i|h_i)$ is a model of the probability of the next word given its history. N-gram models take the view that $p(w_i|h_i)$ depends only on the value of the $N-1$ last words in the history, but some models attempt to extract richer information from h_i; for instance, decision trees over h_i have been used as a basis for constructing probability distributions over w_i.

A powerful approach to constructing history-based models is based on conditional maximum entropy distributions of the form

$$p(w|h) = \frac{1}{Z(h)} \exp \sum_k \lambda_k f_k(h, w),$$

where the f_ks are feature functions of the input-output pair (h, w), the λ_k are the parameters to be trained, and $Z(h)$ is a normalizing term. In some sense that can be made formally precise, such a distribution is the most "neutral" among distributions constrained to preserve the empirical expectations of the f_ks. By adding well-chosen features, one can then force the distribution to be consistent with certain empirical observations. Among features that have been proved practically useful, one finds "skipping bigrams" that model the dependency of w_i relative to w_{i-2}, skipping over w_{i-1}, and "triggers," which generalize caches and model long-range lexical influences (for instance, if *stock* appears somewhere in a document, *bond* is more likely to occur later), but in principle the addition of various other syntactic or semantic features is possible, under the usual caveat that adding too many features may lead to overfitting effects and must be controlled by feature selection procedures or some form of regularization.

History-based maximum entropy models have been reported by some to significantly decrease the perplexity of n-gram models, but other researchers are more cautious, pointing out that combinations of smoothed n-gram and cache often perform at similar levels.

Whole Sentence Maximum Entropy Models

Because the history-based approach models a sentence by predicting one word at a time, phenomena which refer to a whole sentence, such as parsability, global semantic coherence, or even sentence length are at best awkward to model in the approach. In addition, the partition function $Z(h)$, which involves a sum over all the words in the lexicon, has in principle to be computed at decoding time for each

position in the sentence, which is computationally demanding. For these reasons, the following whole-sentence maximum entropy model is sometimes considered:

$$p(s) = \frac{1}{Z}\, p_0(s)\, \exp \sum\nolimits_k \lambda_k f_k(s),$$

where s is a whole sentence, the f_ks are arbitrary features of s, $p_0(s)$ is a baseline distribution (typically corresponding to a standard n-gram model), the λ_k are parameters, and Z is a normalization constant.[11] At decoding time, Z being a constant need not be considered at all and the objective to maximize is a simple linear combination of the features.[12] On the other hand, training is computationally expensive because, *at this stage*, Z does need to be considered (it depends on the λ_ks, which vary during training), and in principle it involves an implicit sum over the space of *all* sentences s. This is infeasible, and approximations are necessary, typically in the form of MCMC (Monte Carlo Markov chain) sampling techniques.

1.4.3 Some Recent Research Trends

Syntactically Structured Language Models

There is a large and well-established body of research on statistical parsing techniques for computational linguistics. Until recently, there have been relatively few approaches to language modeling based on such techniques, in part because the focus in traditional models has been on parsing accuracy rather than on the perplexity of the associated text-generation processes (when they are well-defined), in part because most probabilistic parsing models require the availability of manually annotated treebanks, which are scarce and have limited coverage, and may not be immediately suitable to tasks such as large-scale speech recognition. Two recent language models that use statistical parsing are Chelba and Jelinek (1998) and Charniak (2001), which are both based on a form of stochastic dependency grammar, the former operating in a strict left-to-right manner and trying to predict the next word on the basis of a partial parse for its previous history, the latter assigning probabilities to the immediate descendants of a constituent conditioned on the content of its lexical head (which may be to the right of the descendant, which makes this model non–left to right). Perplexity reductions of up to 25% over a baseline trigram model have been reported, but again such reductions tend to decrease when simple improvements to the baseline are included, such as a cache mechanism.

---

11. Although introduced later in the language model literature than the previous history-based models, these nonconditional maximum entropy models are actually closer to the original formulation of the maximum entropy principle by Jaynes (1957).
12. Note, however, that decoding here means assessing a *complete* sentence s, and that these models are ill-suited for incremental evaluation of sentence *prefixes*.

Topic-Based Modeling and Related Approaches

Topic-based document modeling has been for some time now a hot topic in information retrieval, one of the best-known techniques being latent semantic analysis (LSA) or its probabilistic counterpart (PLSA). Such techniques allow words to be mapped to a real-valued vector in a low-dimensional "topic space," where Euclidian distance between vectors is an indication of the "semantic" proximity between words, as measured by their propensity to appear in lexically related documents. In Bellagarda (1997) these vectorial representations are used in conjunction with n-grams to build language models where the probability of producing a word is conditioned in part by the topical constitution of its history, as summarized by a vector that accumulates the topical contributions of each of the words in the history.

The previous approach is an instance of modeling statistical dependencies that may span over long ranges, such as a whole sentence or even a document. The neural network–based model of Bengio et al. (2003) is another approach that falls in this category. In this model, words are also represented as vectors in a low-dimensional space, and the process of generating texts is seen as one of generating sequences of such vectors. The model learns simultanously the mapping of words to vectors and the conditional probabilities of the next vector given the few previous vectors in the history. As words are "forced" into a low-dimensional vectorial representation by the learning process (in which different occurrences of a word get the same representation), words that show similar contextual behaviors tend to be mapped to vectors that are close in Euclidian space. Recently, similar techniques have been applied to language models in the context of SMT (Déchelotte et al., 2007).

Bayesian Language Modeling

Some recent approaches to document topic-modeling, such as latent Dirichlet allocation (LDA; see Blei et al., 2003) attempt to characterize the problem in strict "Bayesian" terms, that is, in terms of a hierarchical generative process where probabilistic priors are provided for the parameters. Dynamic Bayesian networks is another active area of research which also considers hierarchical time-dependent generative processes which are actually generalizations of hidden Markov models (HMM) with structured hidden layers. These methods are starting to percolate to language modeling, in models that attempt to characterize the production of word sequences through a structured generative process that incorporates a topic-modeling component (Wallach, 2006; Wang, 2005; Mochihashi and Matsumoto, 2006).

Also in the Bayesian tradition are recent attempts to provide "probabilistic-generative" explanations of the Kneser-Ney smoothing procedure in terms of the so-called Chinese restaurant process which is claimed to explain the differential treatment of type counts and occurrence counts in the procedure (Goldwater et al., 2006; Teh, 2006).

Discriminative Language Modeling

As mentioned at the beginning of this section, while *perplexity* as a measure of performance of a language model has the advantage of universality across applications, it is not always correlated with task-related measures of performance, such as the word error rate in speech recognition, or the BLEU or NIST scores in statistical machine translation. In speech recognition, for more than 15 years, this problem has been addressed, not so much in the subtask of language modeling proper, but rather in the so-called acoustic modeling subtask (recovering a word hypothesis from its acoustic realization), where acoustic models have been trained with methods such as maximum mutual information estimation (MMIE) or minimum classification error (MCE), which attempt to learn model parameters with the direct objective of minimizing recognition errors (Huang et al., 2001).

Such discriminative methods have recently gained a large following in all areas of NLP, and especially in statistical machine translation, as witnessed by several chapters in this book (chapters 7, 8, 10, 11). Concerning the use of discriminative models for language modeling proper, a representative paper is Roark et al. (2004), which applies learning based on perceptrons and conditional random fields, in a speech recognition context, to the task of tuning the parameters of a language model (weights of individual n-gram features) on the basis of a training set consisting of input-output pairs where the input is a lattice of word choices returned by a baseline speech-recognition system and the output is the correct transcription, and where the objective is to find parameters that favor the selection of the correct transcription from the choices proposed by the input lattice, as often as possible on the training set. In chapter 6, Mahé and Cancedda introduce another approach to learning a language model discriminatively in the context of machine translation, this time by using kernels rather than explicit features.

1.5 Phrase-Based MT

Phrase-based MT is currently the dominant approach in statistical MT. It incorporates five key innovations relative to the classic approach discussed in section 1.3:

- the use of log-linear models instead of a simple product of language and translation models;
- the use of multiword "phrases" instead of words as the basic unit of translation, within a simple one-to-one generative translation model;
- minimum error-rate training of log-linear models with respect to an automatic metric such as BLEU, instead of maximum likelihood training;
- a clearly defined and efficient heuristic Viterbi beam search procedure; and
- a second *rescoring* pass to select the best hypothesis from a small set of candidates identified during search.

The phrase-based approach is due to Och and Ney (2004). Our presentation in the following sections is loosely based on Koehn et al. (2003), who give a synthesis of Och's method and related approaches by other researchers.

1.5.1 Log-Linear Models

Recall that the noisy-channel approach combines contributions from a language model $p(\mathbf{e})$ and a "reversed" translation model $p(\mathbf{f}|\mathbf{e})$ by multiplying them. A slight generalization of this is to apply exponential weights in order to calibrate the contribution of each model: $p(\mathbf{e})^{\alpha_1} p(\mathbf{f}|\mathbf{e})^{\alpha_2}$. Taking logs and generalizing the language and translation models to arbitrary feature functions $h(\mathbf{f}, \mathbf{e})$ gives a log-linear analog to Eq. (1.1):

$$\begin{aligned} \hat{\mathbf{e}} &= \operatorname*{argmax}_{\mathbf{e}} \sum_i \alpha_i h_i(\mathbf{f}, \mathbf{e}) \\ &\approx \operatorname*{argmax}_{\mathbf{e},\mathbf{a}} \sum_i \alpha_i h_i(\mathbf{f}, \mathbf{a}, \mathbf{e}), \end{aligned} \tag{1.2}$$

where the standard Viterbi approximation on the second line simplifies the search problem and gives features access to the alignment $\mathbf{a}$ connecting $\mathbf{f}$ and $\mathbf{e}$. This framework is more flexible than the original noisy-channel approach because it can easily accommodate sources of information such as bilingual dictionaries which are difficult to incorporate into generative probabilistic translation models. Commonly used features are logs of forward and reversed translation model probabilities and language model probabilities, as well as a simple word count and a phrase distortion model (described in more detail below). A key assumption made by the search procedure is that features decompose linearly; that is, if $(\mathbf{f}, \mathbf{a}, \mathbf{e})$ can be split into a set of disjoint phrase triples $(\tilde{f}_k, a_k, \tilde{e}_k), k = 1 \ldots K$, then $h(\mathbf{f}, \mathbf{a}, \mathbf{e}) = \sum_{k=1}^{K} h(\tilde{f}_k, a_k, \tilde{e}_k)$. This motivates calling the framework log-linear rather than simply linear, since log probabilities have this property, but ordinary probabilities do not. It is also worth noting that $p_\alpha(\mathbf{e}|\mathbf{f}) = \exp(\sum_i \alpha_i h_i(\mathbf{f}, \mathbf{e}))/Z(\mathbf{f})$ can be interpreted as a maximum entropy model for $p(\mathbf{e}|\mathbf{f})$, where $Z(\mathbf{f})$ is a normalizing factor. This was the original formulation of the log-linear approach in Och (2002).

1.5.2 The Phrase-Based Translation Model

The key features used in Eq. (1.2) are related to the *phrase-based* model for $p(\mathbf{e}, \mathbf{a}|\mathbf{f})$.

This model is based on a simple and intuitive generative process: first, $\mathbf{f}$ is segmented into contiguous phrases (word sequences of arbitrary length), then a translation is chosen for each phrase, and finally the resulting target phrases are reordered to form $\mathbf{e}$. Unlike the IBM models, there are no parts of $\mathbf{f}$ or $\mathbf{e}$ that are not covered by a phrase, and each phrase has exactly one translation.

Segmentations are usually assumed to be uniformly distributed, but the other two parts of the generative process—translation and reordering—each give rise to log-linear features. Let $\tilde{e}_1 \ldots \tilde{e}_K$ be a segmentation of $\mathbf{e}$ into phrases, and $\tilde{f}_1 \ldots \tilde{f}_K$ be the

corresponding source phrases (i.e., the phrases in $\mathbf{f}$, in the order their translations appear in $\mathbf{e}$). Then a "reversed" translation feature can be defined by assuming that phrases are generated independently:

$$h_T(\mathbf{f}, \mathbf{a}, \mathbf{e}) = \sum_{k=1}^{K} \log p(\tilde{f}_k | \tilde{e}_k).$$

A "forward" translation feature can be defined analogously using $p(\tilde{e}_k | \tilde{f}_k)$. Koehn et al. (2003) propose a simple distortion feature for capturing reordering[13]:

$$h_D(\mathbf{f}, \mathbf{a}, \mathbf{e}) = \sum_{k=1}^{K} -|\text{begin}(\tilde{f}_k) - \text{end}(\tilde{f}_{k-1}) - 1|,$$

where $\text{begin}(\tilde{f})$ and $\text{end}(\tilde{f})$ are the initial and final word positions of $\tilde{f}$ in $\mathbf{f}$ (with $\text{end}(\tilde{f}_0) = 0$). This assigns a score of 0 to translations which preserve source phrase order, and penalizes displacements from the "expected" position of the current source phrase (immediately after the preceding phrase) by the number of words moved in either direction.

The phrase-translation distributions $p(\tilde{f}|\tilde{e})$ and $p(\tilde{e}|\tilde{f})$ are defined over a set of phrase pairs called a *phrase table*. Phrase-table induction from parallel corpora is crucial to the performance of phrase-based translation. It typically proceeds by first word-aligning the corpus, then, for each sentence pair, extracting all phrase pairs that are compatible with the given word alignment, under the criterion that a valid phrase pair must not contain links to words outside the pair. For example, in the sentence pair: *Je suis heureux / I am very happy*, with word alignment *Je/I, suis/am, heureux/very_happy*, legal phrase pairs would include *Je/I*, *Je suis/I am*, and *heureux/very happy*, but not *heureux/happy*. In general, this algorithm is fairly robust to word-alignment errors, which tend to affect recall more than precision.

The existence of a standard phrase-extraction algorithm independent of the underlying word alignment has stimulated interest in improved word-alignment techniques. As mentioned in section 1.3, the baseline approach of Och and Ney (2003) relies on IBM model (typically IBM 4) alignments carried out from each translation direction, then *symmetrizes* them into a single alignment by beginning with links in their intersection, then selectively adding links from their union, according to various heuristics. Recently proposed alternatives include combining alternate tokenizations (see chapter 5 by Elming, Habash and Crego), the use of enhanced IBM models (He, 2007), more principled symmetrization techniques (Liang et al., 2007), discriminative techniques (Blunsom and Cohn, 2006), and semisupervised techniques (Fraser and Marcu, 2006), to name only a few.

One difficulty in judging alignment quality for SMT is that the ideal metric—performance of the resulting MT system—is very expensive to compute. In a recent

13. This equation assumes that begin() and end() have access to $\mathbf{a}$, which maps the permuted source phrases $\tilde{f}_1 \ldots \tilde{f}_K$ to their positions in $\mathbf{f}$.

paper, Fraser and Marcu (2007) argue against the use of the popular *alignment error rate* metric as a stand-in, and propose an alternative which correlates better with MT performance.

Once phrase pairs have been extracted from the training corpus, it remains to estimate the phrase-translation distributions $p(\tilde{f}|\tilde{e})$ and $p(\tilde{e}|\tilde{f})$. These may be obtained directly as relative frequencies from joint counts of the number of times each phrase pair was extracted from the corpus. Compositional estimates based on lexical probabilities from the IBM models or word-alignment counts are often used in addition to relative frequencies (Koehn et al., 2003; Zens and Ney, 2004). It is interesting that the heuristic method outlined in the previous paragraphs for populating phrase tables and estimating conditional phrase probabilities seems to perform better than more principled generative algorithms (e.g., Marcu and Wong, 2002) for estimating these distributions. DeNero et al. (2006) argue that this is essentially due to the inclusive property of considering alternative segmentations simultaneously (e.g., learning both *Je/I, suis/am*, and *Je suis/I am* in the example above) rather than forcing segmentations to compete as would estimation with the EM algorithm.

1.5.3 Minimum Error-Rate Training

Given an automatic metric as discussed in section 1.2—for instance, BLEU—minimum error-rate training seeks the vector of log-linear parameters $\hat{\alpha}$ that optimize the metric on a training corpus:

$$\hat{\alpha} = \underset{\alpha}{\text{argmax}}\ \text{BLEU}(\hat{E} = \underset{E}{\text{argmax}}\ \log p_\alpha(E|F),\ R), \tag{1.3}$$

where $\log p_\alpha(E|F) = \sum_{(\mathbf{e},\mathbf{f})\in(E,F)} \log p_\alpha(\mathbf{e}|\mathbf{f})$. The inner argmax is a search with log-linear model $\log p_\alpha$, applied to a source-language corpus F to find the best translation $\hat{E}$. The outer argmax finds the α for which $\hat{E}$ maximizes BLEU with respect to a reference translation R. Och (2003) showed that this approach produces models that score better on new corpora according to the chosen metric than does a maximum likelihood criterion.

Eq. (1.3) is difficult to optimize because the inner argmax is very expensive to compute, and also because BLEU is a nondifferentiable function of α. The standard solution, proposed in Och (2003), is to approximate the inner argmax with a maximization over a small set of n-best candidate translations for F (on the order of 100 translations per source sentence). This makes it fast enough that general optimization techniques can be applied to solve the outer argmax. The success of this approximation depends on being able to identify n-best lists that are representative of the entire search space. Och does this by iterating over different values of $\hat{\alpha}$, each time using the new value of $\hat{\alpha}$ to add new candidates to the n-best lists, which are in turn used to update $\hat{\alpha}$. Bad values of $\hat{\alpha}$ will therefore add bad candidates to the lists which will allow the optimization to avoid these values in future iterations. The complete algorithm is:

1. Initialize $\hat{\alpha}$ and set the n-best set $B = \emptyset$.
2. Find $B(\hat{\alpha})$, the n-best translations for each source sentence according to $p_{\hat{\alpha}}$.
3. Set $B = B \cup B(\hat{\alpha})$. Stop if B doesn't change.
4. Set $\hat{\alpha} = \underset{\alpha}{\operatorname{argmax}}\ \mathrm{BLEU}(\hat{E} = \underset{E \in B}{\operatorname{argmax}}\, p_\alpha(E|F),\ R)$ and go to step 2.

Since the number of hypotheses produced by the decoder is finite, this algorithm is guaranteed to terminate. In practice, it converges fairly quickly, usually after ten iterations or so.

The optimization in step 4 may be solved using Powell's algorithm (Press et al., 2002). This is a general optimization algorithm that iteratively chooses lines $\alpha + \gamma\alpha'$ which must be optimized in the scalar value γ by means of a user-supplied "subroutine." Since $\log p_\alpha$ is linear in α, the score it assigns to each hypothesis in an n-best list is linear in γ. There are therefore at most $n - 1$ values of γ at which BLEU can change for a single n-best list, and at most $m(n - 1)$ values for a set of m n-best lists. By examining the intervals between these points, it is possible to efficiently obtain an exact solution to the problem of maximizing BLEU as a function of γ.

The bottleneck in Och's algorithm is the decoding operation in step 2. This makes it impractical for use on training corpora larger than about 1000 sentences, which in turn limits the number of log-linear parameters that can be reliably learned. Also, the ability of Powell's algorithm to find a good optimum appears to degrade with larger parameter sets (Och et al., 2004), so the typical number of parameters is on the order of ten. Och (2003) also proposes a smoothed version of BLEU which would allow gradient-based techniques to be used instead of Powell's algorithm, but it is not clear whether this approach would give better performance with large feature sets.

Alternatives to Och's algorithm use a different strategy for solving the central problem of costly decoding time: modify the decoder to work faster, typically by considering only monotone alignments (i.e., ones in which source and target phrases have the same order), and by using an aggressive pruning threshold. If decoding is fast enough, the outer argmax in Eq. (1.3) can be solved directly with a general optimization algorithm, e.g., downhill simplex (Zens and Ney, 2004) or simultaneous perturbation stochastic approximation (Lambert and Banchs, 2006). These approaches appear to be competitive with Och's. They have the advantage that they can optimize any parameter of the decoder, rather than just log-linear model weights, but the disadvantage of making poor estimates for features that are sensitive to monotonic decoding, for instance distortion.

Other approaches to minimum error-rate training include recent efforts to train very large sets of parameters, such as weights for Boolean phrase-pair features defined over the phrase table, on very large corpora. Liang et al. (2006) iterate the following: generate n-best lists for the corpus, controlling decoder speed by using a limited-distortion model (neighbor swaps only) and limiting to short sentences, then use the perceptron algorithm to update toward the best candidate in each n-

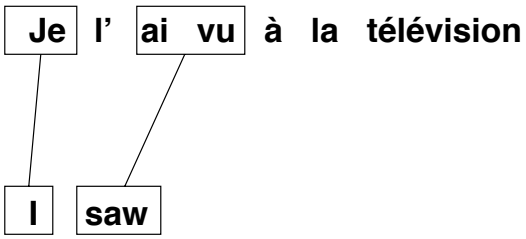

Figure 1.2 A partial hypothesis during decoding, including its alignment. This is extended by choosing a phrase that matches part of the source sentence with no alignment connection (the *uncovered* part), for instance, *à la*, and appending one of its translations from the phrase table, for instance *on*, to the target prefix, giving in this case the new prefix *I saw on*.

best list. Tillmann and Zhang (2006) iterate the following: decode and merge 1-best translations with existing n-best lists, controlling decoder speed by limiting distortion as above, then use stochastic gradient descent to minimize a "margin-inspired" distance function between n-best candidates and oracle translations generated by using the references to guide the decoder. These approaches give only fairly modest gains, possibly because of sacrifices made for decoder efficiency, and possibly because performance appears to be rather insensitive to the exact values of the phrase-translation parameters $p(\tilde{f}|\tilde{e})$.

1.5.4 Search

As we have seen, the problem of decoding Eq. (1.2) is central to minimum error-rate training, and of course in all applications of statistical MT as well. It is NP-complete for phrase-based MT, as it is for the IBM models, but somewhat simpler due to the one-to-one restriction on phrase translation. The standard Viterbi beam search algorithm (Koehn, 2004a) builds target hypotheses left to right by successively adding phrases. As each phrase is added to a hypothesis, the corresponding source phrase is recorded, so the complete phrase alignment is always known for all hypotheses, as illustrated in figure 1.2. Search terminates when the alignments for all active hypotheses are complete, i.e. when all words in the source sentence have been translated. At this point, the hypothesis that scores highest according to the model is output.

A straightforward implementation of this algorithm would create a large number of hypotheses: for each valid segmentation of the source sentence, and each bag of phrases created by choosing one translation for each source phrase in the segmentation, there would be one hypothesis for each permutation of the contents of the bag. Even when the phrase table is pruned to reduce the number of translations available for each source phrase, the number of hypotheses is still unmanageably huge for all but the shortest source sentences. Several measures are used to reduce the number of active hypotheses and the space needed to store them.

First, hypotheses are *recombined*: if any pair of hypotheses are indistinguishable by the model in the sense that extending them in the same way will lead to the same change in score, then only the higher-scoring one needs to be extended. Typically, the lower-scoring one is kept in a *lattice* (word graph) structure, for the purpose of extracting n-best lists (Ueffing et al., 2002) once search is complete. The conditions for recombination depend on the features in the model. For the standard features described above, two hypotheses must share the same last $n-1$ words (assuming an n-gram LM), they must have the same set of covered source words (though not necessarily aligned the same way), and the source phrases aligned with their last target phrase must end at the same point (for the distortion feature).

Recombination is a factoring operation that does not change the results of the search. It is typically used in conjunction with a pruning operation that *can* affect the search outcome. Pruning removes all hypotheses whose scores fall outside a given range (or *beam*) defined with respect to the current best-scoring hypothesis; or, in the case of *histogram pruning*, fall below a given rank.

Three strategies are used to make the comparison between hypotheses as fair as possible during pruning. First, the scores on which pruning is based include an estimate of the *future score*—the score of the suffix required to complete the translation—added to the current hypothesis score. If future scores are guaranteed never to underestimate the true suffix scores, then they are *admissible*, as in A* search, and no search errors will be made. This is typically too expensive in practice, however. Each feature contributes to the future score estimate, which is based on analyzing the uncovered portion of the source sentence. The highest-probability translations from the phrase table are chosen, and are assigned LM scores that assume they can be concatenated with probability 1 (i.e., the LM scores only the inside of each target phrase), and distortion scores that assume they are arranged monotonically. Phrase table and language model future scores can be precomputed for all subsequences of the source sentence prior to search, and looked up when needed.

Comparing two hypotheses that cover different numbers of source words will tend to be unfair to the hypothesis that covers the greater number, since it will have a smaller future score component, and since future scores are intentionally optimistic. To avoid this source of bias, hypotheses are partitioned into equivalence classes called *stacks*, and pruning is applied only within each stack. Stacks are usually based on the number of covered source words, but may be based on their identities as well, in order to avoid bias caused by source words or phrases that are particularly difficult to translate.

Along with recombination and pruning, a final third used to reduce the search space is a limit on the distortion cost for two source phrases that are aligned to neighboring target phrases. Any partial hypotheses that cannot be completed without exceeding this limit are removed. Interestingly, imposing such a limit, typically seven words, often improves translation performance as well as search performance.

There are a number of ways to arrange the hypothesis extension and pruning operations described in the preceding paragraphs. A typical one is to organize the search according to stacks, as summarized in the following algorithm from Koehn (2004a):

- Initialize stack 0 with an empty hypothesis.
- For each stack from 0...J-1 (where J is the number of source words):
  - For each hypothesis g in the stack:
    - For each possible extension of g, covering j source words:
      - Add the extension to stack j, checking for recombination.
      - Prune stack j.
- Output the best hypothesis from stack J.

There have been several recent improvements to the basic Viterbi search algorithm. Huang and Chiang (2007) propose *cube pruning*, which aims to reduce the number of expensive calls to the language model by generating hypotheses and performing an initial pruning step prior to applying the language model feature. Moore and Quirk (2007) use an improved distortion future score calculation and an early pruning step at the point of hypothesis extension (before LM calculation). Both techniques yield approximately an order of magnitude speedup.

1.5.5 Rescoring

The ability of the Viterbi search algorithm to generate n-best lists with minimal extra cost lends itself to a two-pass search strategy in which an initial log-linear model is used to generate an n-best list, then a second, more powerful, model is used to select new best candidates for each source sentence from this list in a *rescoring* (aka *reranking*) pass.

The advantage of this strategy is that, unlike the candidates considered during the first pass, the candidates in an n-best list can be explicitly enumerated for evaluation by the model. This means that there is virtually no restriction on the kinds of features that can be used. Examples of rescoring features that would not be practical within the first-pass log-linear model for decoding include long-range language models, "reversed" IBM 1 models for $p(f|\mathbf{e})$, and features that use IBM 1 to ensure that all words have been translated. Och et al. (2004) list many others.

To assess the scope for improvement due to rescoring, one can perform an oracle calculation in which the best candidates are chosen from the n-best list with knowledge of the reference translations.[14] This gives impressive gains, even for fairly short n-best lists containing 1000 candidates per source sentence. However, this is

14. For metrics like BLEU, which are not additive over source sentences, this can be approximated by choosing the candidate that gives the largest improvement in global score, and iterating.

somewhat misleading, because much of the gain comes from the oracle's ability to game the automatic metric by maximizing its matching criterion, and is thus not accessible to any reasonable translation model (or even any human translator). In practice, gains from rescoring are usually rather modest—often barely statistically significant; by the time n-best lists have been compiled, most of the damage has been done.

Rescoring is most often performed using log-linear models trained using one of the minimum-error techniques described in section 1.5.3. Alternatives include perceptron-based classification (learning to separate candidates at the top of the list from those at the bottom) and ordinal regression (Shen et al., 2004); and also Yamada and Muslea's ensemble training approach, presented in chapter 8.

1.5.6 Current Status

Phrase-based translation remains the dominant approach in statistical MT. However, significant gains have recently been achieved by syntactic methods (particularly on difficult language pairs such as Chinese-English; see section 1.6), by factored methods, and by system combination approaches (see section 1.7).

1.6 Syntax-Based SMT

While the first SMT models were word-based, and the mainstream models are currently phrase-based, we have witnessed recently a surge of approaches that attempt to incorporate syntactic structure, a movement that is reminiscent of the early history of rule-based systems, which started with models directly relating source strings to target strings, and gradually moved toward relating syntactic representations and even, at a later stage, logical forms and semantic representations.

The motivations for using syntax in SMT are related to consideration of fluency and adequacy of the translations produced:

- Fluency of output depends closely on the ability to handle such things as agreement, case markers, verb-controlled prepositions, order of arguments and modifiers relative to their head, and numerous other phenomena which are controlled by the syntax of the target language and can only be approximated by n-gram language models.
- Adequacy of output depends on the ability to disambiguate the input and to correctly reconstruct in the output the relative semantic roles of constituents in the input. Disambiguation is sometimes possible only on the basis of parsing the input, and reconstructing relative roles is often poorly approximated by models of reordering that penalize distortions between the source and the target word orders, as is common in phrase-based models; this problem becomes more and more severe when the source and target languages are typologically remote from each other (e.g.,

subject-verb-object languages such as English, subject-object-verb languages such as Japanese, or languages that allow relatively "free" word order such as Czech).

There are many approaches to incorporating syntax in SMT systems, of which we will describe only a few representative instances. One dimension that is useful for organizing the different approaches is the extent to which they assume some form of a priori knowledge about the syntactic structure of the source and target languages. While certain approaches require that external parsers exist for both the source and the target languages, and use parsed bilingual corpora for training their models, some approaches only require that such parsers exist for the source or for the target language,[15] while some more radical approaches do not require any externally given parser but learn aligned structured representations on the basis of an unparsed bilingual corpus. We start with these "resource-poor" approaches and move gradually toward the more "resource-intensive" ones.

1.6.1 Parser-Free Approaches

Currently probably the most representative among the parser-free approaches is Chiang (2005)'s *hierarchical phrase-based translation*. The model is in line with previous work by Wu (1997) on *inversion transduction grammars* for parsing bilingual corpora and is formally based on a generalization of these grammars, namely *synchronous context-free grammars*. Such grammars are bilateral context-free grammars that simultaneously describe constituents in a source and in a target language and have rules such as (source language is Chinese here)

$$X \rightarrow \langle X \text{ zhiyi, one of } X \rangle,$$

where the source and target expressions on the right-hand side contain terminals and "coupled" nonterminals that correspond to subconstituents which are in translation correspondence. These rules may be automatically extracted from word-aligned phrase pairs by identifying nonterminals with aligned subphrases.

One important restriction in the formalism used by Chiang is that there is only a single generic nonterminal type X, in contrast to externally motivated grammars, which would have nonterminals such as noun phrase (NP), verb phrase (VP), and so forth. Under this constraint, rules such as the above can be seen as direct generalizations of standard biphrases, where the coupled Xs correspond to sub-

15. Another aspect that distinguishes systems is whether they are tree to string, string to tree, or tree to tree, but this aspect is not as clear as the dimension involving reference to external parsers; a system that only uses an external parser for the source can still technically be tree to string or tree to tree, in the latter case through *projecting* trees from the source side of the bilingual corpus over to the target side and using the structural correspondences thus found; a similar remark is true of systems that only involve external parsers on the target side.

biphrases of these biphrases, which were themselves in translation correspondence in the training corpus and have been "anonymized" into X.

Decoding is performed by parsing the source side with the synchronous grammar and simultaneously producing a target parse. Competing derivations are scored according to a log-linear model whose weights are learned based on a minimun-error training procedure.

1.6.2 Parser on the Target

An early attempt to use syntax in SMT was presented by Yamada and Knight (2001), who considered a model for translating from Japanese to English. They use the Collins parser for English for building tree structures over the target side of the bilingual corpus and then learn a mapping from an English tree to a Japanese string through a sequence of transformations: first the nodes of the English tree are locally reordered, then some Japanese words (typically function words) are inserted in the reordered English tree, then the remaining English words are translated into Japanese words, and finally a Japanese string is produced. At training time, EM is used in order to learn the parameters of the different transformations that maximize the likelihood of the training set, and the resulting set of probabilistic transformations constitutes the "translation model" part of a noisy-channel model (hence the model is indeed eventually used for translating from Japanese to English, and not the reverse.) While the model is sometimes described as mapping Japanese strings to English trees (hence as a string-to-tree model), from the description it is clear that internally, Japanese trees are actually built; however, these Japanese trees are not obtained by reference to an independent parser of Japanese, but rather as a kind of projection of externally motivated English parses.

More recently, researchers from the same research group at the Information Sciences Institute have applied powerful formalisms, known as *tree-to-string transducers*, to relate target trees with source strings. In Marcu et al. (2006), such a model is used to translate from Chinese to English. When applied in reverse to the source string (such formalisms can be used either in a forward or reverse direction), the tree-to-string transducer behaves similarly to a context-free grammar (meaning that chart-parsing techniques can be applied to factorize the derivations above a given Chinese string) but each derivation can be seen as a recipe for gluing together English tree "stumps" and producing a complex English parse tree; thus the correspondence between derivations on the source side and trees on the target side is not as direct as in synchronous tree grammars and allows more flexible couplings. At decoding time the application of rules is under the control of a log-linear model that combines features computed on derivations, and the model weights are learnt by mininum error training. The system was claimed in 2006 to be the first

to show BLEU improvements over phrase-based models on experiments conducted over large-scale, open domain translation tasks.[16]

1.6.3 Parser on the Source

An instance of using an external parser on the source only is the work conducted at Microsoft Research by Quirk et al. (2005), who use an in-house rule-based parser, NLPWIN, that produces dependency structures for English. Given a bilingual English-French training corpus, word aligned with GIZA++,[17] the source dependency trees are projected onto the French side of the corpus and from the aligned sentence-level dependency structures obtained, a collection of aligned "treelets" is extracted. These treelets are structural analogs to the biphrases extracted in phrase-based SMT and are constrained to be connected subcomponents of the dependency structure, but not necessarily to project onto contiguous subspans of the word string. At decoding time, the source sentence is parsed, is decomposed into treelets, and a target representation is constructed by gluing together the associated target treelets, under the control of log-linear features. An important aspect of the model (permitted by the use of dependency structures) is that the target representations thus obtained are underdetermined with regard to the relative order of the dependents of a head. This order is determined by a separate model, which is independently trained; this separation of work between treelet training and order training gives flexibility to the model, as the extracted treelets themselves do not need to encapsulate word-ordering considerations.

In chapter 7, Wellington, Turian, and Melamed present another instance where an externally trained parser (Bikel's parser, trained on the Penn treebank) is used to parse the English source side of a bilingual English-French corpus and where projection techniques are used to obtain parallel trees in the target language; however the focus here is on a generic training technique for learning how to transduce a source tree into a target tree and could probably be applied as well to a situation where the target trees were obtained by an independent external parser. Training works by attempting to reconstruct the corpus target trees from the corpus source trees through a sequence of atomic decisions that incrementally build nodes of the target tree, given both the context of the source and the context of previous decisions. The training procedure interleaves feature selection actions and parameter tuning actions, using a boosted ensemble of decision trees under an l_1 regularization regime that favors sparse features.

16. This claim was based on experiments for Chinese-English in the NIST-06 campaign, and continued in NIST-08 for the same language pair. However in the case of Arabic-English, phrase-based systems still win in the later campaign.

17. Even if not mentioned explicitly, the use of GIZA++ for word-aligning a bilingual corpus is actually a shared prerequisite of most of the approaches described in this section.

1.6.4 Parsers on the Source and Target

One approach in which structural a priori knowledge of both the source and the target languages plays an important role was introduced by Cowan et al. (2006). They consider translation from German to English, and use the Dubey parser for German and a modification of the Collins parser for English in order to parse both sides of a bilingual Europarl corpus. The English parser produces structures of a specific form, aligned extended projections (AEPs), which are inspired by the formalism of lexicalized tree adjoining grammar (Frank, 2002). The focus of the paper is to learn the translation of German clauses into English clauses, as opposed to full sentences, and the AEP of an English clause can be seen as a syntactic template to which a sequence of elementary operations have been applied, such as selecting active or passive voice, instantiating the subject slot, choosing the inflection of the verb, etc. The order of such operations is linguistically motivated, for instance the inflection of the verb may depend on the subject. After the bilingual corpus has been parsed on both sides, aligned German clausal structures and English clausal AEPs are extracted, and the goal of training is to learn a sequence of decisions that will permit reconstruction of the English AEP from the German structure. The first such decision is choosing the English syntactic template, then the following decisions correspond to the elementary operations that determine the AEP. Each decision is performed on the basis of features of the German structure and of the previous decisions taken, and the training of the associated weights is done through a structured perceptron procedure. At decoding time, a beam-search procedure is applied, which attempts to find the sequence of decisions which has the largest score according to these weights. In this approach we see a clear instance where a careful linguistic design (nature and order of the elementary operations leading to an AEP) is explicitly exploited for organizing the learning procedure.

1.7 Some Other Important Directions

Statistical machine translation is a very active field of research, and the chapters in this book illustrate a range of promising directions. It would be impossible to cover all ongoing efforts: in this section we briefly touch on some that we perceive as particularly interesting.

1.7.1 Factored Models

The majority of published research on machine translation reports experiments on language pairs having English as target. Translating into other languages requires solving problems that are just negligible in English. Morphology, for instance, is very simple in English compared to most other languages, where verbs can have tens of alternative forms according to mood, tense, etc.; nouns can have different forms for nominative, accusative, dative, and so on. Dictionaries for such

languages tend to be much larger (empirical linguists speak of a lower *token/type ratio*), and reliable statistics are harder to gather. Moreover, when translating from a morphologically poor language (e.g., English) into a morphologically rich one (e.g., Russian), purely word- or phrase-based models can have a hard time, since generating the appropriate morphology might require rather sophisticated forms of analysis on the source: n-gram-based language models can only go so far.

Koehn and Hoang (2007) introduced *factored translation models*, where source words are enriched with linguistic annotation (e.g., lemmas, parts of speech, morphological tags). Separate distributions model translation from source lemmas to target lemmas and from source parts of speech and morphology to their target equivalent. A deterministic morphological generator, finally, combines target lemmas and morphological information to reconstruct target surface forms (i.e., actual words).

Factored language models, where words are represented as bundles of features and the conditioning history can be composed of heterogeneous elements (e.g., a word and a parts of speech), were introduced earlier (Bilmes and Kirchhoff (2003)). The use of *factored word-sequence kernels* in discriminatively-trained language models (chapter 6) falls in the same line of work.

1.7.2 Model Adaptation

The quality of translation and language models depends heavily on the amount of training data. Training corpora of sufficient size for a given language pair, domain, and genre might not be readily available: one is thus left with the uncomfortable choice of either training on few data points coming from the distribution of interest (*on-topic corpora*), or on many data points from a different distribution (*off-topic corpora*). Language model adaptation has been extensively investigated, especially in conjunction with speech recognition. The interest in translation model adaptation, on the other hand, is more recent.

Hildebrand et al. (2005) proposed information retrieval techniques to select from a training set sentence pairs whose source is similar to a given test set, and train only on those. Munteanu et al. (2004) went further, and proposed a classifier for identifying sentences which are a translation of one another in a *comparable corpus* (i.e., a set of similar documents in two different languages).

More recently, Foster and Kuhn (2007) introduced a method based on mixture models: the training data is divided into different components, models (both translation and language) are trained separately on each component, and are combined at translation time with weights which depend on the similarity of the source document and the training data of each component.

Similarly, Xu et al. (2007) train separate language models on different domains, and also use limited on-topic parallel data to re-estimate the weights of the features of a log-linear model. When a new document needs translation, it is first categorized into one domain and then translated using the adapted language model and feature weights.

1.7.3 System Combination

System combination techniques aim to exploit the diversity in translation outputs from different MT systems in order to improve over the best single system. A challenge in doing so is that alternate translations may have very different surface properties such as lexical choice and word order, making it difficult to blend them into a single reasonable output. Recently, Rosti et al. (2007b) proposed an effective solution that consists in choosing one system's hypothesis to establish the word order of the output. The other hypotheses are aligned to this *skeleton* using edit distance, resulting in a constrained word lattice known as a *confusion network* from which the output is derived by searching with a language model and weighted scores from the input systems. Chapter 13 by Matusov, Leusch and Neyin this book extends this approach using IBM alignments rather than edit distance for aligning hypotheses. System combination techniques have recently improved to the point where they reliably give gains over the best single system, even when the other participating systems are relatively weak.

1.7.4 Kernel Methods for Machine Translation

A rather radical departure from existing approaches to SMT is proposed by Wang et al. (2007) (see also chapter 9). Using kernels on strings it is possible to map separately sentences of the source and of the target language into distinct vector spaces (or *feature spaces*). Conceptually the translation problem can thus be decomposed into

1. mapping a source language sentence into a vector in the input feature space;
2. mapping this vector into a vector in the output feature space by means of an appropriate function;
3. mapping a vector from the output feature space into a target language sentence.

The function in the second step can be learned from a training set using an appropriate regression algorithm (such as ridge regression). In practice, the first and the second steps are conflated in that a kernel is used to implicitly map source sentences into the input feature space. The third step, the *inverse image* problem, can be very hard, depending on the kernel used on the target side.

1.8 Machine Learning for SMT

The preceding sections suggest that training a statistical machine translation system is very closely related to supervised learning. At the core of the statistical approach to MT is the attempt to map some input source language sentences $\mathbf{f}$ to some output $\mathbf{e}$. There are, however, a number of idiosyncracies which preclude the straightforward application of known machine learning techniques to SMT. In this

final section, we will relate SMT to various standard machine learning frameworks, and discuss the issue of learning with an uncertain loss, as well as the issue of dividing the MT learning problem into smaller manageable problems, as opposed to adopting an end-to-end approach.

1.8.1 Translation as a Learning Problem

In the context of translation, the output variable—a target language sentence—is formally a discrete variable. In machine learning, predicting a discrete variable usually leads to a classification framework. However, SMT clearly does not fit comfortably in this context: the output space, although discrete, has too many modalities and too much structure. The regression framework is not a much better fit: the output space is not continuous and is very unsmooth, as sentences with similar surface forms may have very different meanings (and therefore translations). In fact, MT is closer to the relatively recent framework of learning with structured output (Taskar, 2004; Tsochantaridis et al., 2005).

The work presented in chapter 9 in this book is a typical example of such an approach. Input and output data are projected into two vector spaces using the implicit mappings $\Phi(\mathbf{f})$ and $\Psi(\mathbf{e})$ provided by a kernel operating on structured data (in that case, sentences in the source and target languages). A multivariate regression model $\Psi(\mathbf{e}) \approx \mathbf{W}\Phi(\mathbf{f})$ may then be used to model the dependencies between the projected input and output, even though the original data is highly structured and does not live in vector spaces. One challenge of this approach is the *preimage problem*: given an estimate $\widehat{\Psi} = \widehat{\mathbf{W}}\Phi(\mathbf{f})$ for a new source sentence $\mathbf{f}$, which target sentence $\widehat{\mathbf{e}}$ should be chosen, such that its image through the implicit mapping, $\Psi(\widehat{\mathbf{e}})$, is "closest" to the regression estimate $\widehat{\Psi}$? This is a very difficult problem for most kernels operating on structured data, and very closely corresponds to the *decoding* step in the traditional SMT framework.

Further work will no doubt appear along those lines. In fact machine translation is a natural field of application for machine learning techniques operating on structured inputs and outputs, as large amounts of training data are available, for a variety of language pairs (e.g., Koehn, 2005; Steinberger et al., 2006). In fact, another important challenge for structured learning methods is to scale up to the corpus sizes commonly used in statistical machine translation, where millions of sentence pairs are not unusual (see, e.g., chapter 8).

It may also be interesting to draw a parallel with the ranking framework, which has been addressed in machine learning in the context of information retrieval, collaborative filtering, extractive summarization, or multiclass categorization, among others. Traditionally, machine translation has been defined as the problem of producing *one* correct translation $\mathbf{e}$ for each source sentence $\mathbf{f}$. However, an arguably equally efficient alternative would be to seek an ordered list of target hypotheses $\mathbf{e}^{(1)}, \mathbf{e}^{(2)}, \ldots$, such that correct translations are placed above incorrect ones. This may be relevant in two situations:

1. When there are many correct translations of the source sentence, all of them, not a single one, should be placed at the top of the list.

2. When the model is unable to generate any correct translation, it still makes sense to try to rank nearly correct hypotheses at the top of the list.

In the traditional approach to SMT, such as described in section 1.5, ranking is actually used in at least two ways. First, decoders based on log-linear models usually output ordered n-best lists of translation hypotheses rather than a unique, most probable translation. Second, an additional reranking step, using, for example, more and more complicated feature functions, is used to improve the n-best list by promoting "correct" hypotheses to the top of the list. In both situations, however, ranking is typically based on the output of a model trained for discrimination, not for ranking. Theoretical and empirical results in machine learning (e.g., Cortes and Mohri, 2004) suggest that models trained to minimize an error rate may not provide optimal ranking performance, especially for uneven distributions and high error rates, which is precisely the situation of most MT systems. Placing MT in the framework of a ranking problem and using techniques designed to optimize the ranking performance therefore seems like an interesting direction of investigation. This is in fact the appraoch presented by Shen et al. (2004) for the rescoring stage, and they obtain encouraging results using perceptron-based ordinal regression.

1.8.2 Learning with an Inaccurate Loss

One aspect that crucially sets machine translation apart from most other applications of machine learning is the issue of evaluation. As explained in section 1.2, even when reference translations are available, there is no exact way to calculate, or even define, the cost associated with a new translation hypothesis. This is at odds with most areas where machine learning has been applied. Indeed, most machine learning techniques, at their core, attempt to minimize some loss or risk associated with the prediction. What can be done when such a loss is not available? One typical strategy is to target a different, approximate loss, work on that instead, and hope for the best.

Standard approaches to SMT such as word-based models (section 1.3) rely on maximizing the likelihood on the training set. Within the state-of-the-art framework of phrase-based SMT (section 1.5), phrase tables and language models are typically estimated using word or phrase counts, which corresponds to maximum likelihood estimates, possibly with the addition of some smoothing or regularization. However, the likelihood is not necessarily a good indicator of translation quality. As the unattainable reference of translation evaluation is human judgment, the reliability of the various metrics described in sections 1.2 is usually assessed by their correlation

with human evaluation. The need to optimize these metrics[18] has led to *minimum error-rate training* (section 1.5), where some model parameters are trained by directly minimizing one metric. In the context of machine learning, gradient descent has been used to optimize differentiable losses. More recent work has been targeted to optimizing kernel machines on metrics such as the F-score used in information retrieval or the area under the curve (AUC) used for ranking (Callut and Dupont, 2005; Joachims, 2005). A challenging avenue for future research would be to train some of the newly proposed SMT techniques that depart from the log-linear models by directly optimizing the MT metrics, instead of relying on the standard losses such as the squared error.

An additional consideration is that automatic MT metrics focus on different aspects of the difference between the hypothesis and reference translations: n-gram precision, recall, edit distance, bag-of-word similarity, etc. Arguably, none of these is sufficient to fully account for the difference between two sentences. However, they may conceivably account for *some* of the difference. It would therefore be interesting to consider optimizing not just a single metric, but several of these metrics simultaneously. Techniques from multiobjective, or multicriteria, optimization (Steuer, 1986) may be relevant to that purpose. One of the simplest ways to do that is to combine the multiple objective functions into a single *aggregate objective function* (Giménez and Amigó, 2006). The system may then be optimized on the aggregate measure, in order to increase reliability and robustness.

Finally, the situation of MT evaluation suggests a more speculative question. Is it possible to set up a framework for learning with an imprecisely defined loss? In machine translation, we have a number of approximate losses which have measurable correlation with the "real," unknown loss. By learning on those, we surely learn something about the underlying task, provided the correlation is positive. By contrast, overtraining on the approximate metric will likely degrade the performance on the real loss. It seems to us that this is not a very commonly studied setting in machine learning. However, advances in that direction would certainly have the potential to benefit research in statistical machine translation.

1.8.3 End-to-End Learning for SMT

Current statistical translation systems involve a combination of several models (translation, language model, log-linear model, rescoring; section 1.5). The parameters associated with each of these are usually estimated more or less independently, leading to a *highly stratified* parameter estimation: the parameters of the translation model are derived from the phrase table using essentially maximum likelihood parameters; the language model parameters are obtained by smoothing the maxi-

18. Besides the quest for better translation quality, one additional motivation is that international MT evaluations are usually carried out using automatic MT evaluation metrics. Optimizing the right metric can have a direct impact on a system's ranking.

mum likelihood (or minimum perplexity) estimates; parameters of the phrase-based translation model and rescoring model are usually estimated using minimum error-rate training, etc. In addition, parts of the model, such as the distortion feature function (section 1.5.2), are parameterless, but could conceivably be made more flexible with the addition of a few parameters.

The obvious limitation of this approach is that the overall model is divided into smaller parts, each optimized locally on a loss that may be only loosely related to the overall translation goal. Instead, one would ideally like to optimize all model parameters globally, on the overall loss. Note, however, that in the context of machine translation, this means optimizing over millions of parameters of the translation and language models, in addition to the log-linear parameters. Recent advances in discriminative training of machine translation models have started addressing this issue. This is the case for two approaches described at the end of section 1.5.3. Tillmann and Zhang (2006) propose a new translation model and an associated discriminative training technique that optimizes millions of parameters using a global score (such as BLEU). Liang et al. (2006) also propose an *end-to-end* approach relying on a perceptron trained on millions of features, but which also includes translation and language model probabilities as features, thus retaining part of the stratification in the model estimation. In both cases, the models differ substantially from the current state of the art of phrase-based translation.

The issue of stratified vs. end-to-end parameter estimation therefore suggests (at least) two directions for improving translation performance. One would be to limit the stratification of current phrase-based models by estimating more parameters globally. The second is obviously to improve recent end-to-end models, which are currently competitive only with baseline versions of phrase-based models (usually a fairly standard Pharaoh system), but not with the more evolved versions used, for example, in international evaluations.

1.9 Conclusion

In this introduction, we have given an overview of current statistical machine translation techniques. We also provide pointers to the literature for readers wishing to acquire more information on specific topics. Our hope is that this chapter is self-contained and broad enough for the reader not especially familiar with SMT to now turn to and benefit from the more advanced topics addressed in the following chapters of this book.

Appendix: On-line SMT Resources

- Statistical machine translation resources (`http://www.statmt.org/`): includes links to the yearly workshop on machine translation

- Moses (`http://www.statmt.org/moses/`): SMT system implementing phrase-based translation and factored model, with beam-search decoder
- Pharaoh (`http://www.isi.edu/publications/licensed-sw/pharaoh/`): freely available decoder for phrase-based SMT
- GIZA++ (`www.fjoch.com/GIZA++.html`): toolkit implementing the IBM models
- SRILM (`http://www.speech.sri.com/projects/srilm/`): widely used SRI language modelling toolkit
- LDC (Linguistic Data Consortium, `http://www.ldc.upenn.edu/`): provider of multilingual data
- ELDA (Evaluations and Language Resources Distribution Agency, `http://www.elda.org/`): operational body of the European Language Resources Association and provider of multilingual data
- Europarl (`http://www.statmt.org/europarl/`): parallel corpus including 11 European languages
- NIST (`http://www.nist.gov/speech/tests/mt`): the annual MT evaluation carried out by NIST